Chinese Journal of Aeronautics, (2017), 30(1): 264–273

Chinese Society of Aeronautics and Astronautics
& Beihang University

Chinese Journal of Aeronautics

cja@buaa.edu.cn
www.sciencedirect.com

A novel surface mesh deformation method for handling wing-fuselage intersections

Mario Jaime Martin-Burgos [a], Daniel González-Juárez [a], Esther Andrés-Pérez [b,c,*]

[a] *Fluid Dynamics Branch, National Institute for Aerospace Technology (INTA), Ctra. de Ajalvir, km. 4.5, 28850 Torrejón de Ardoz, Spain*
[b] *Engineering Department, Ingeniería de Sistemas para la Defensa de España (ISDEFE-INTA), Ctra. de Ajalvir, km. 4.5, 28850 Torrejón de Ardoz, Spain*
[c] *Technical University of Madrid (UPM), Ronda de Valencia 3, 28012 Madrid, Spain*

Received 24 November 2015; revised 22 January 2016; accepted 6 August 2016
Available online 22 December 2016

KEYWORDS

Aerodynamic shape optimization;
Computational fluid dynamics;
Computer-assisted design;
Geometry parameterization;
Mesh deformation;
Non-Uniform Rational B-Splines

Abstract This paper describes a method for mesh adaptation in the presence of intersections, such as wing-fuselage. Automatic optimization tools, using Computational Fluid Dynamics (CFD) simulations, face the problem to adapt the computational grid upon deformations of the boundary surface. When mesh regeneration is not feasible, due to the high cost to build up the computational grid, mesh deformation techniques are considered a cheap approach to adapt the mesh to changes on the geometry. Mesh adaptation is a well-known subject in the literature; however, there is very little work which deals with moving intersections. Without a proper treatment of the intersections, the use of automatic optimization methods for aircraft design is limited to individual components. The proposed method takes advantage of the CAD description, which usually comes in the form of Non-Uniform Rational B-Splines (NURBS) patches. This paper describes an algorithm to recalculate the intersection line between two parametric surfaces. Then, the surface mesh is adapted to the moving intersection in parametric coordinates. Finally, the deformation is propagated through the volumetric mesh. The proposed method is tested with the DLR F6 wing-body configuration.

1. Introduction

In shape optimization problems,[1–3] the baseline geometry is modified in the search of an optimal shape. In each of the geometry modifications, it is required to update the mesh, and this can be achieved by using an automatic remeshing process. In the context of aircraft design, the grid generation of complex configurations involving several components is usually an expensive and time-consuming task that requires great expertise. In order to avoid the regeneration of the

* Corresponding author at: Engineering Department, Ingeniería de Sistemas para la Defensa de España (ISDEFE-INTA), Ctra. de Ajalvir, km. 4.5, 28850 Torrejón de Ardoz, Spain.
E-mail address: eandres@isdefe.es (E. Andrés-Pérez).

Peer review under responsibility of Editorial Committee of CJA.

http://dx.doi.org/10.1016/j.cja.2016.12.026
1000-9361 © 2016 Chinese Society of Aeronautics and Astronautics. Production and hosting by Elsevier Ltd.

computational grid, automatic mesh deformation techniques[4,5] are considered a fast approach for small deformations, which is commonly employed in automatic optimization loops and aero-elastic simulations.

However, surface mesh deformation methods still suffer several limitations in the presence of moving intersections, such as wing-fuselage and wing-pylon-nacelle assemblies. Without a proper treatment of intersections, the use of automatic optimization methods for aircraft design is limited to individual components. In addition, once these components are assembled, the aerodynamic properties might be significantly different because of the fluid interaction between them.

In general, the intersection curve between two Non-Uniform Rational B-Splines (NURBS) surfaces cannot be determined analytically. There have been several attempts in the literature to address this problem. T-splines are designed to deal with trimming NURBS, although they might present difficulties to represent a watertight curve intersection.[6] The term "watertight" connotes no unwanted gaps or holes. The surface/surface intersection between NURBS is tackled in the research of Sederberg et al.[7] by using a moving algorithm: first, a local unit step direction H is determined by intersecting the tangent planes of the two surfaces, and guessing a new approximation $P_1 = P_0 + LH$, where L is determined from an adaptive method. P_1 is approximated from a point at the intersection P_0 and the direction tangent to both planes. In addition, in the research done by Gagnon and Zingg,[8] the geometries are defined analytically with watertight networks of surfaces and the approach is applied to a lift-constrained drag minimization of a conventional regional jet. Moreover, in the work of Hwang and Martins,[9] the approach is to model an aircraft as a union of untrimmed surfaces (i.e. surfaces with four topological edges). Regarding surface mesh deformation, different approaches are proposed to use spring/mass type idealization[10–12] or solving elasticity equations[13–15]. In this work, an adaptation based on the Laplacian field is suggested.

This method takes advantage of the CAD definition, in the form of NURBS surfaces, to recalculate the intersection, and therefore, requires deforming the NURBS along with the grid. This parametrization is widely supported by software tools, but for optimization applications, the final shape is strongly conditioned by the number and distribution of the control points. NURBS extracted directly from the CAD-file are unlikely suitable for optimization. Thus, a new NURBS needs to be generated, which still represents the original geometry within acceptable error margins, which is a time-consuming task that requires a great deal of expertise.

The above issue does not appear using differentiable volumetric methods, such as Free Form Deformation (FFD),[16] and its extension to volumetric B-splines control box.[17] The intersection between components is accurately calculated in each optimization step, while at the same time, the CAD file is preserved to easily share the geometry between software applications (for instance, in case of coupled fluid–structure optimization problems). After the computation of the intersection line, the surface mesh vertices are deformed by following their NURBS parametric coordinates, which have been previously obtained from the mesh generation application or calculated with an appropriate inversion point technique.[18,19] Finally, the surface grid is updated to match the moving intersection with a mesh deformation algorithm. Once the surface grid is properly adapted to the new configuration, a volumetric adaptation is employed to build the new computational grid.

The control box extends the FFD concept, using NURBS basis. This technique requires the additional effort of calculating the parametric coordinates from the spatial coordinates through an appropriate point inversion algorithm. However, the control box approach has important advantages over FFD, such as deformation locality, arbitrary setup of the control points, selection of the smoothness and the ability to choose the order of the interpolation, while achieving the same pleasing deformation characteristics as surface NURBS. Actually, the conventional FFD can be considered a subset of control box.

Additionally, some parameterizations can fuse components into the same description, so wing-fuselage surfaces are treated as a single entity with all the deformation being continuous and the intersection naturally adapted. However, there is an important advantage of describing specific components with a unique set of NURBS. Different aircraft components, such as wing, fuselage, nose and pylon, require different skills and expertise, and it is generally convenient to keep them intact while one component is optimized. For example, modifications of the wing should not modify the fuselage geometry. Without an underlying geometry, provided by NURBS, global deformations might result in unwanted modifications of other components.

This paper is structured as follows: the next section briefly introduces the mathematical background of NURBS. Then, Section 3 describes the proposed mesh adaptation strategy, giving details on the inversion point, intersection recalculation and surface deformation algorithms. Finally, the proposed strategy is applied to three different deformation scenarios (bump, rotation and displacement movements of the wing) of the DLR F6 wing-body configuration and an analysis of the performance and mesh quality metrics are provided in order to validate the approach. The DLR-F6, is a simplified wing-fuselage geometry which has been used for the validation of CFD codes at the AIAA sponsored Drag Prediction Workshops.

2. Mathematical background: Brief introduction to NURBS

NURBS are a standardized geometric description frequently employed by CAD applications to represent a surface skin. By incorporating the NURBS in the design loop, the effort to exchange information in a suitable format between different disciplines, such as aero-dynamic/structural analysis and post-processing tools, is significantly reduced.[8,9] The aerodynamic surface of an aircraft cannot be usually defined with a continuous shape for the whole geometry, and therefore, several NURBS patches have to be employed to assemble the different sections defining intersections and continuity conditions.

From the mathematical point of view, NURBS surfaces[20] are parametric representations defined as

$$S(\xi,\eta) = \frac{\sum_i^N \sum_j^M U_i^p(\xi) V_j^q(\eta) w_{ij} C_{ij}}{\sum_i^N \sum_j^M U_i^p(\xi) V_j^q(\eta) w_{ij}} \quad (1)$$

where $\{\xi,\eta\}$ are the parametric coordinates, U and V the basis functions of orders p and q respectively, C_{ij} the control points, and w_{ij} the weights. One of the most effective methods to calculate the basis functions is through a recursive algorithm, which in the literature is referred to as the De Boor's algorithm[21]

$$\begin{cases} U_i^p(t) = \begin{cases} 1 & \text{if} \quad u_i \leqslant u_{i+1} \\ 0 & \text{otherwise} \end{cases} \\ U_i^p = \dfrac{(t-u_i)U_i^{p-1}(t)}{u_{i+p-1}-u_i} + \dfrac{(u_{i+p}-t)U_{i+1}^{p-1}(t)}{u_{i+p}-u_{i+1}} \end{cases} \tag{2}$$

The terms u_i are coefficients from the so called knot vector, which is a sequence of real numbers that frequently have multiplicity at the beginning and the end of $\{\underbrace{0,\ldots,0}_{p+1},\ldots,\underbrace{1,\ldots,1}_{p+1}\}$ to ensure that the extremes are not ill-defined and t is the parameter, which range from 0 to 1.

The link between the computational grid, employed for Computational Fluid Dynamics (CFD) simulations, and the CAD geometry, defined by NURBS patches, requires the knowledge of the parametric coordinates $\{\xi, \eta\}$ of each surface vertex. With this information, it is possible to recalculate the spatial coordinates of the vertex, given a displacement of the NURBS control points. The calculation of the parametric coordinates from the spatial ones is usually referred to as the so-called inversion point problem.

3. Proposed method for surface mesh deformation with intersections handling

3.1. Flowchart of proposed strategy

The proposed strategy for surface mesh deformation comprises several steps (Fig. 1).

First, the CAD geometry is extracted from the Initial Graphics Exchange Specification (IGES) file, as a collection of several NURBS patches. Then, the parametric coordinates of the surface grid points are calculated using the inversion point algorithm explained in Section 3.2 (Notice that the points in the intersection line have two pairs of parametric coordinates, one for each intersecting NURBS panel). Then, the deformation is applied to the NURBS and therefore the intersection between NURBS panels (in parametric coordinates) $\{\xi^*, \eta^*\}$ has to be recalculated, as detailed in Section 3.3. After that, the parametric coordinates $\{\xi', \eta'\}$ of the surface grid points are updated to match the new intersection, using the deformation algorithm proposed in Section 3.4. From the updated parametric coordinates, the Cartesian coordinates of the surface grid points $\{x', y', z'\}$ are calculated. Finally, the surface deformation is propagated to the volumetric grid, using a conventional deformation algorithm. In this work, the DLR's TAU deformation module[22] has been used. The unsteady TAU-Code solves the compressible, three-dimensional Reynolds-Averaged Navier–Stokes equations using a finite volume formulation. The TAU-Code is based on a hybrid unstructured-grid approach, which makes use of the advantages of semi-structured prismatic grids in the viscous shear layers near walls, and the flexibility in grid generation offered by tetrahedral grids in the surrounding flow volume. The TAU-Code consists of several different modules, including the deformation module, which propagates the deformation of surface grid points to the surrounding volume grid.

3.2. Calculating parametric coordinates of surface grid points (Inversion point algorithm)

The inversion point algorithm calculates the parametric coordinates from the Cartesian coordinates of each surface grid point; $\mathbf{R}^3(x, y, z) \rightarrow \mathbf{R}^2\ (\xi, \eta)$. This is required when this information is not provided from the mesh generation software or when mesh refinement techniques are involved. The parametric coordinates of a surface vertex P are calculated as the projection Q^* to the surface S; understanding projection as the surface point of minimum distance is illustrated in Fig. 2, while Q^0 is an initial arbitrary approximation point on the surface.

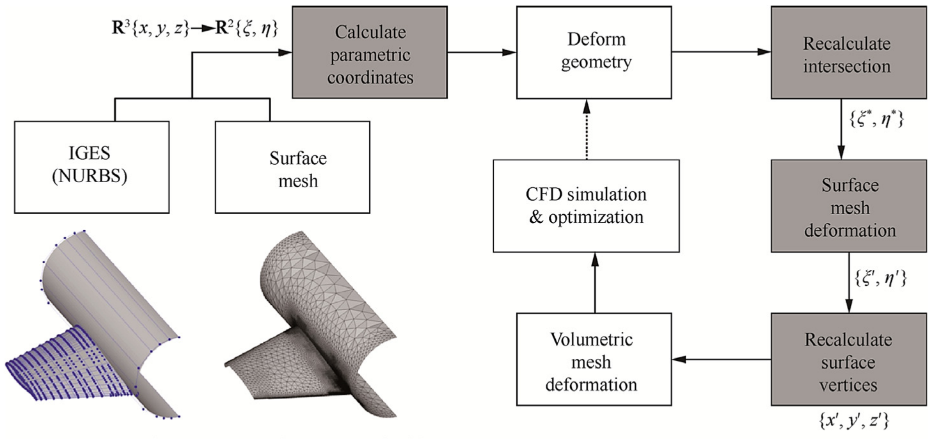
The goal is to find the parametric coordinates of Q^* by solving the following expression

$$\min(|P - S(\xi, \eta)|^2) \tag{3}$$

where P is a point to be projected, not necessarily on the surface. Provided with a suitable initial estimation $\{\xi, \eta\}^0$, the parametric coordinates of the projection can be efficiently computed with a Newton–Raphson algorithm:

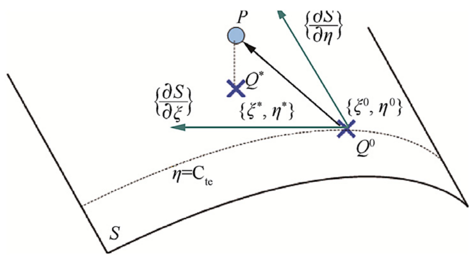
$$\{\xi, \eta\}^{n+1} = \{\xi, \eta\}^n - \frac{f}{f'} \tag{4}$$

To simplify the notation, we call $Q = S\ (\xi, \eta)$. The objective function and its derivatives are


$\mathbf{R}^3\{x, y, z\} \rightarrow \mathbf{R}^2\{\xi, \eta\}$
Calculate parametric coordinates
Deform geometry
Recalculate intersection
$\{\xi^*, \eta^*\}$
IGES (NURBS)
Surface mesh
CFD simulation & optimization
Surface mesh deformation
$\{\xi', \eta'\}$
Volumetric mesh deformation
Recalculate surface vertices
$\{x', y', z'\}$

Note: Grey boxes represent the core steps in this strategy.

Fig. 1 Flowchart of proposed approach.



P
$\left\{\frac{\partial S}{\partial \eta}\right\}$
$\left\{\frac{\partial S}{\partial \xi}\right\}$
Q^*
$\{\xi^*, \eta^*\}$
$\{\xi^0, \eta^0\}$
Q^0
η=C$_{te}$
S

Fig. 2 Scheme of inversion point as a projection from Cartesian space to parametric surface S.

$$\begin{cases} f = |P - S(\xi,\eta)|^2 = |P - Q|^2 \\ f' = \begin{bmatrix} f_\xi \\ f_\eta \end{bmatrix} = \begin{bmatrix} 2\dfrac{\partial S(\xi,\eta)}{\partial \xi}(P - Q) \\ 2\dfrac{\partial S(\xi,\eta)}{\partial \eta}(P - Q) \end{bmatrix} \end{cases} \tag{5}$$

where f is the Euclidian distance and f' is the Jacobian matrix, with respect each parametric direction. Multiplying the denominator in Eq. (4) by its transpose and simplifying lead to the following iterative algorithm

$$\begin{cases} \Delta\xi = \dfrac{\left[(P - Q^i) * \frac{\partial S(\xi,\eta)}{\partial \xi}\right] \cdot \left|\frac{\partial S(\xi,\eta)}{\partial \xi}\right|^2 - \left[(P - Q^i) * \frac{\partial S(\xi,\eta)}{\partial \eta}\right] \cdot \left[\frac{\partial S(\xi,\eta)}{\partial \xi} * \frac{\partial S(\xi,\eta)}{\partial \eta}\right]}{d} \\ \Delta\eta = \dfrac{\left[(P - Q^i) * \frac{\partial S(\xi,\eta)}{\partial \eta}\right] \cdot \left|\frac{\partial S(\xi,\eta)}{\partial \eta}\right|^2 - \left[(P - Q^i) * \frac{\partial S(\xi,\eta)}{\partial \xi}\right] \cdot \left[\frac{\partial S(\xi,\eta)}{\partial \xi} * \frac{\partial S(\xi,\eta)}{\partial \eta}\right]}{d} \\ d = \left|\dfrac{\partial S(\xi,\eta)}{\partial \xi}\right|^2 \cdot \left|\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right|^2 - \left[\dfrac{\partial S(\xi,\eta)}{\partial \xi} * \dfrac{\partial S(\xi,\eta)}{\partial \eta}\right]^2 \end{cases} \tag{6}$$

where the symbol * denotes the dot product, while $\cdot$ the conventional scalar product. Notice that $|a|^2 = a^* a$. The above expression can be interpreted as the projection of P to the plane tangent to the surface at Q and it is a general expression for parametric surfaces (not limited to NURBS).

One critical aspect is to provide a suitable initial value for the algorithm to converge. One approach consists of calculating the projection of the vertex to the NURBS control polygon[19]; this is the surface formed by the control points or the equivalent second order NURBS surface. NURBS basis are monotone positive functions, but knots, kinks, edges and other discontinuities might create convergence problems.

3.3. Treatment of intersection

Surface grid points at joints and intersections belong to more than one NURBS surfaces. For instance, in a wing-body configuration, the geometry of the wing is represented by the NURBS surface S_a, while the geometry of the fuselage is represented by S_b. Vertices at the intersection are represented by two pairs of parametric coordinates $\{\xi_a, \eta_a\}$ and $\{\xi_b, \eta_b\}$, one for each NURBS. At the intersection of two surfaces S_a and S_b, the parametric coordinates satisfy $S_a(\xi_a, \eta_a) = S_b(\xi_b, \eta_b)$. The intersection is obtained by solving the following three non-linear equations with four unknowns.

$$S_a(\xi,\eta) - S_b(\xi,\eta) = 0 \tag{7}$$

In this particular problem, the intersection curve is not required, but the parametric coordinates $\{\xi_a, \eta_a\}$ and $\{\xi_b, \eta_b\}$ for all vertices at the intersection are required, once the intersection has moved. To simplify the notation, let us call $P = S_a(\xi_a, \eta_a)$ and $Q = S_b(\xi_b, \eta_b)$, the spatial coordinates of the vertex calculated from the first and second pair of parametric coordinates respectively (Fig. 3).

After a movement on the surface, the original parametric values at the intersection $\{\xi_a, \eta_a\}^0$ and $\{\xi_b, \eta_b\}^0$ no longer correspond to the same spatial location, because the intersection is now represented by a different set of parametric coordinates. For relatively small deformations, the new parametric coordinates can be efficiently computed with the previous iterative algorithm employed for the inversion point algorithm in Eq. (6). Equivalently, the intersection can be found as the projection of P on S_b and Q on S_a. For numerical robustness, the denominator in Eq. (6) is $2d$ (instead of just d). Given starting points P^0 and Q^0, the strategy is to alternate the projection of P and Q at each iteration:

$$\begin{cases} \{\xi_a, \eta_a\}^1 = \{\xi_a, \eta_a\}^0 - \dfrac{f(\xi_a^0, \eta_a^0, P^0, Q^0)}{f'(\xi_a^0, \eta_a^0, P^0, Q^0)} \\ P^1 = S_a(\xi_a^1, \eta_a^1) \\ \{\xi_b, \eta_b\}^1 = \{\xi_b, \eta_b\}^0 - \dfrac{f(\xi_b^0, \eta_b^0, P^1, Q^0)}{f'(\xi_b^0, \eta_b^0, P^1, Q^0)} \\ Q^1 = S_b(\xi_b^1, \eta_b^1) \\ \ldots \end{cases} \tag{8}$$

At each iteration, the above algorithm projects the line $\overrightarrow{PQ}$ onto the plane defined by the derivatives, until it finds the intersection. After the deformation, the initial values can be the original ones. However, robust solutions are also obtained by approximating the surface grid vertices displaced by the deformation, which does not provide an exact solution, but it is closer than the original vertex coordinates.

3.4. Surface mesh deformation

Once the parametric coordinates of the surface vertex at the intersection are recalculated, the surface grid should be adapted to the new configuration. A perturbation of a Laplacian field, after linearization, can be expressed with the following equation

$$a' = \frac{\sum_i (x_i - x_i^0) \cdot \phi(x_i, a)}{\sum_i \phi(x_i, a)} + a^0 \tag{9}$$

In the above expression, a is the vertex parametric coordinates, which is connected to x_i nodes (Fig. 4).

The notation x^0 indicates the original position of the node, while the term $\phi(x, a)$ is an arbitrary constant weight function. One value that works well is the inverse of the square of the Euclidean distance between the nodes.

$$\phi(x_i, a) = |x_i^0 - a^0|^{-2} \tag{10}$$

In this way, deformations are mostly absorbed by the big elements of the grid, while small elements, such as those at the boundary layer, remain rigid. The system of equations generated can be solved iteratively with a Jacobi algorithm until the residual converges to zero $(x_i^{t+1} - x_i^t) \to 0$ as follows

$$a^{t+1} = \frac{\sum_i (x_i^{t+1} - x_i^0) \cdot \phi(x_i, a)}{\sum_i \phi(x_i, a)} + a^0 \tag{11}$$

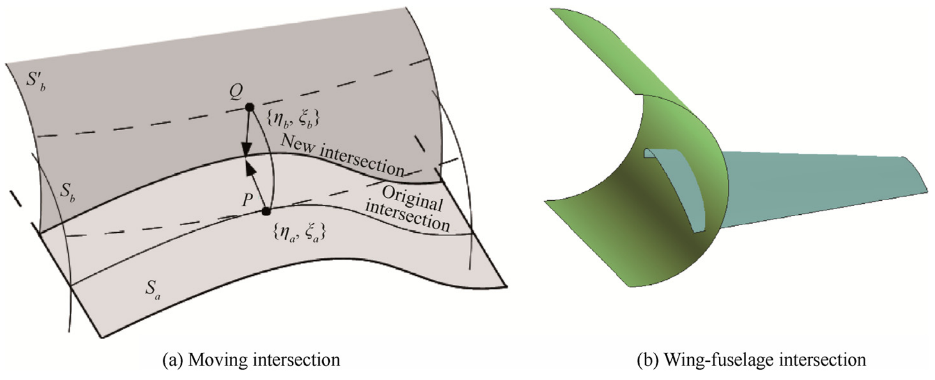


Fig. 3 Notation of recalculating parametric coordinates.

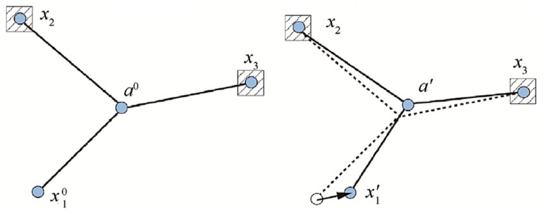


Fig. 4 Scheme of displacement of vertex parametric coordinates a due to displacement of x_1.

This method is very easy to implement and fully explicit without requirement of inverting a large matrix system. In this scheme, each parametric coordinate is solved independently, so the deformation algorithm is performed twice, one for ξ and one for η. Eq. (11) resembles an algorithm to solve the Laplace equation. Because the solution of the Laplace equation is unique, the above algorithm always converges to a unique state, independently of the starting values or the order of the computations.

By performing the adaptation on parametric coordinates, surface vertices are ensured to be on the geometric CAD definition, defined by the NURBS patches. Fig. 5 shows how the adaptation performs in parametric coordinates in the wing-fuselage test case.

The complete strategy is summarized in the following algorithm: (see Table 1).

Table 1 Summary of proposed methodology for surface mesh deformation with intersections handling.

```
Inputs: grid, geometry (IGES format)
Output: grid, geometry (IGES format)

#inversion point
For each surface vertex that belongs to the NURBS panel $S_a$ and $S_b$
provide an initial estimation $\{\xi, \eta\}^0$
compute Newton–Raphson algorithm as indicated in Eq. (4) to obtain $\{\xi, \eta\}$
End

#recalculate intersection
For each surface vertex that belongs to the intersection
recalculate its parametric coordinates as indicated in Eq. (8), so $S_a(\xi_a, \eta_a) = S_b(\xi_a, \eta_a)$
End

#surface deformation
For each surface vertex that belongs to the NURBS panel $S_a$
    deform the mesh (in parametric coordinates) with a deformation algorithm, as indicated in Eq. (11);
    use the new intersection previously calculated as a boundary.
    idem with each surface vertex that belongs to the NURBS panel $S_b$
    recalculate the Cartesian coordinates $\{x,y,z\} = S(\xi, \eta)$ with the new parametric coordinates
End for
```

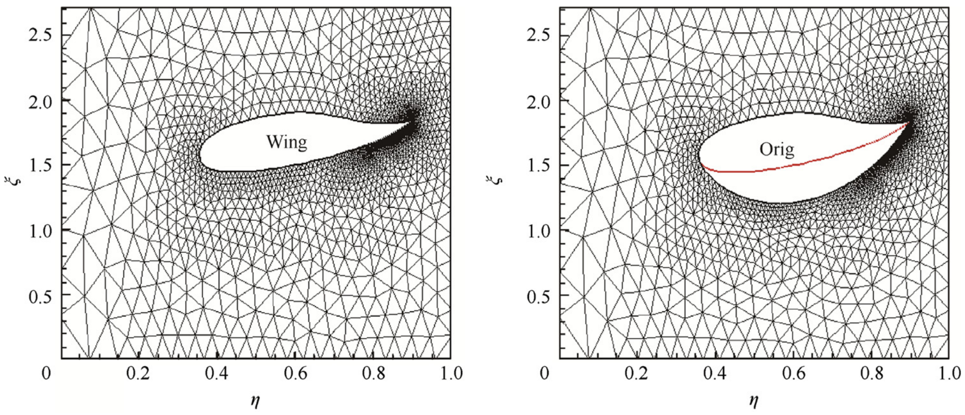


Fig. 5 Example of adaptation of fuselage vertex parametric coordinates upon a deformation of wing using presented deformation algorithm.

3.5. Considerations on scheme differentiation for aerodynamic shape optimization

In this particular approach, the design variables are the movement of the control points. Gradients can be calculated via finite differentiation, which is mathematically simple but computationally expensive, as it requires one simulation for each design variable. The advantage of finite differences is that gradients can be calculated as a black box including the effect of the moving intersection.

Alternatively, the adjoint method is more efficient to compute the gradients, as the number of runs is essentially independent from the number of design variables. In the continuous adjoint approach, gradients are computed as the integration over the surface, which turns into a summation over the finite surface elements of the computational grid:

$$\delta J = \sum_{S} \delta j \cdot (\delta \boldsymbol{x} \cdot \boldsymbol{n}) \mathrm{d}s$$

where δJ are the gradients calculated on the design variables, δj are the sensitivities provided by the adjoint to each surface vertex, $\delta \boldsymbol{x}$ are the geometric sensitivities (that is, how the surface is modified by a movement of the design variable), $\boldsymbol{n}$ is the surface normal, and $\mathrm{d}s$ is the element dual area associated to the vertex. At the intersection, there is geometric discontinuity, where both geometric sensitivity and normal are undetermined and gradients might also be inaccurate,[3,23] since there are tangential terms that are not included in the above formulation.[23] Although the effect of those vertices might be "diluted" in the integration, the effect of the gradients can be estimated as the combined effect of two dummy vertices very close to the intersection (one for each intersecting surface).

4. Numerical results

In order to validate the performance of the proposed approach, it was applied to the DLR-F6 wing-body aircraft configuration, which also includes a nacelle/pylon component (Fig. 6). This configuration was used in the 3rd AIAA CFD Drag Prediction Workshop[24] and the IGES geometry definition considered in this paper has been downloaded from that workshop website.

In this test case, the wing geometry is defined with 7 NURBS patches, the pylon geometry is defined with 20 NURBS patches and the central fuselage is defined with 1 patch. The surface grid is composed of 56.322 vertices and 112.644 triangle elements.

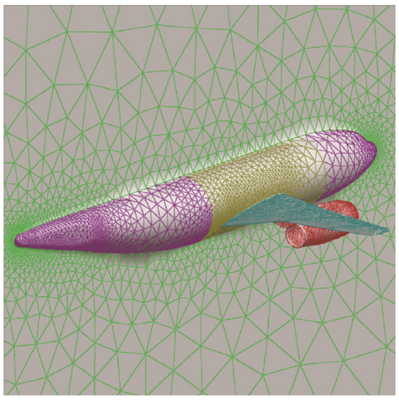


Fig. 6 DLR F6 wing-body-pylon-nacelle configuration.

Three different scenarios (bump deformation, rotation and displacement of the wing) are addressed (Fig. 7). In the first scenario, the wing is displaced in the forward direction along the fuselage. Then, in the second scenario, the upper side of the wing that intersects with the fuselage is modified with a bump deformation. Finally, in the third scenario, the wing is rotated. The fuselage remains intact in all the deformation scenarios.

4.1. Test case scenario number 1: Wing displacement along fuselage

In the first scenario, the wing is displaced in the forward direction along the fuselage (Fig. 8). It can also be observed how, after this displacement, the computational grid is strongly damaged and therefore needs to be fixed. The surface grid close to the junction between the fuselage and the wing is required to be adapted.

Figs. 9 and 10 show how the surface grid is fixed using the presented methodology. Notice that the deformation is restricted to the panel that represents the central section of the fuselage.

4.2. Test case scenario number 2: Bump deformation on upper side of wing

In the second deformation scenario, the patch that represents the upper side of the wing section is modified with a large bump deformation, while the fuselage remains intact. Fig. 11 shows the baseline and deformed configurations.

Mesh deformation is then applied to both fuselage and wing surface grids, although the deformation is more severe at the fuselage and therefore requires more extensive rearrangement. For illustration purpose, only the fuselage grid is shown. Figs. 12 and 13 show the surface grid adaptation.

4.3. Test case scenario number 3: Wing rotation

In the third deformation scenario, the wing is rotated while the fuselage remains intact. Fig. 14 shows the baseline and deformed configurations.

Mesh deformation is then applied to both fuselage and wing surface grids, although the deformation is more severe at the fuselage and therefore requires more extensive rearrangement. For illustration purpose, only the fuselage grid is shown.

Figs. 15 and 16 show the surface grid adaptation by employing NURBS.

4.4. Mesh quality analysis after deformation

In order to complete the validation of the proposed strategy, the quality of the deformed grids has been computed and compared to the original grid. Table 2 shows the comparison of several grid quality metrics (minimum, maximum and mean

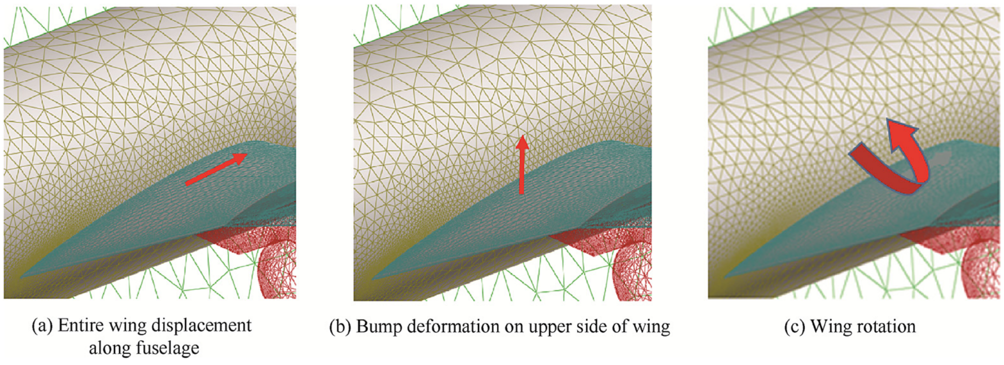

(a) Entire wing displacement along fuselage
(b) Bump deformation on upper side of wing
(c) Wing rotation

Fig. 7 Deformation scenarios considered for validation of proposed approach.

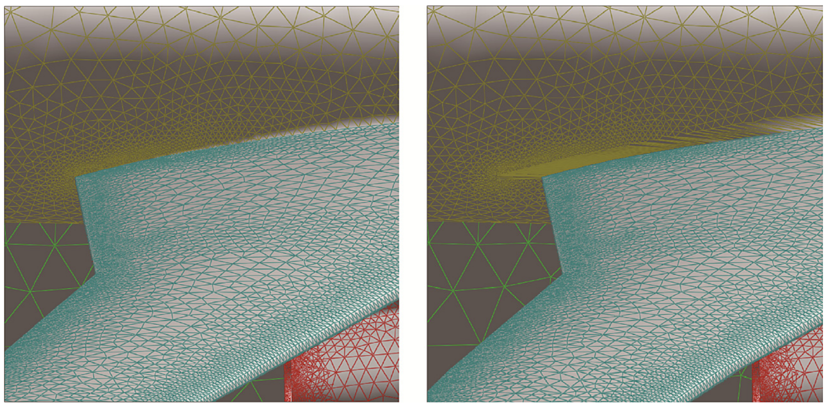


Fig. 8 Displacement of wing in forward direction.

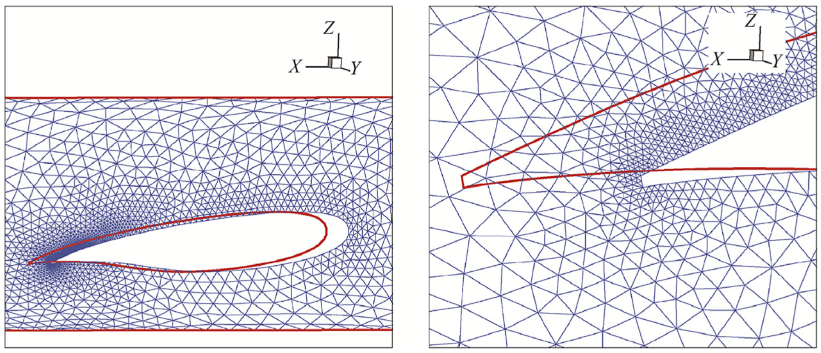


Fig. 9 Grid adaptation upon displacement of wing with NURBS (left) and details of grid adaptation (right).

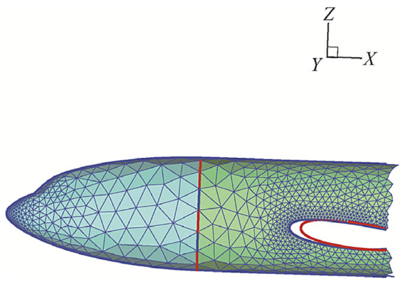


Fig. 10 Adaptation of surface grid with NURBS.

q value, and minimum dihedral angle) for each of the three deformation scenarios, together with the values on the baseline grid. It can be observed that the minimum grid quality value is reduced in all the deformation scenarios, especially in the wing rotation. In addition, the rotation shows a strong reduction of the minimum dihedral angle, which means that this deformation is the most critical one in terms of maintaining grid quality. However, the mean quality remains acceptable for all the deformation scenarios.

The histogram of the quality metric intervals with respect to the number of mesh elements is displayed in Fig. 17, showing a similar distribution between the original grid and the three deformed ones.

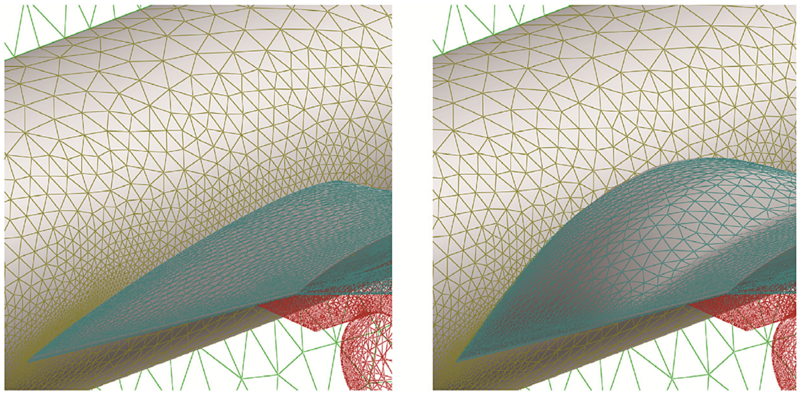


Fig. 11 Deformation of upper side of wing in the second scenario.

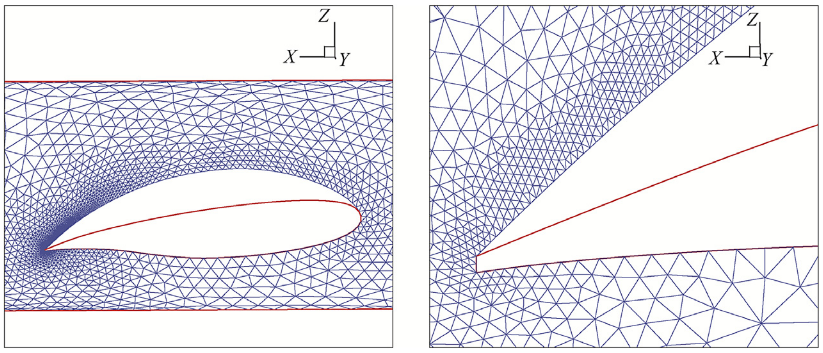


Fig. 12 Surface grid adaptation upon deformation of upper side of wing by using NURBS (left) and details near trailing edge (right).

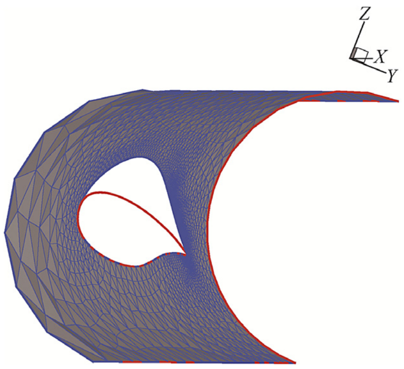


Fig. 13 Grid adaptation using NURBS based strategy has preserved underlying fuselage geometry.

Finally, Table 3 shows the number of mesh elements in each of the grid quality intervals. Although the quality of the deformed grid is acceptable in all cases, it can be seen that in the wing rotation scenario, the number of elements with quality between 0.21 and 0.25 increases from 0 to 5, and most significantly, the elements with quality between 0.31 and 0.35 increase from 2 to 23.

5. Conclusions and further work

The ability to adapt a computational grid upon deformations of the geometry is a useful technique that avoids the need to rebuild the mesh from the scratch in every optimization step. When several components are involved, such as wing-fuselage or

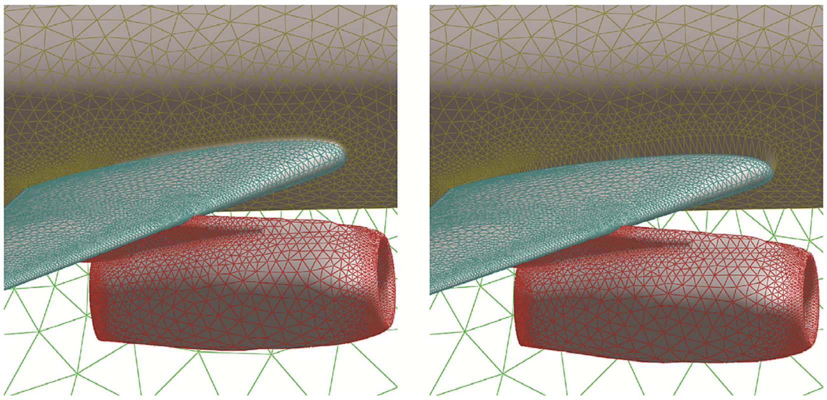


Fig. 14 Deformation of upper side of wing in the third scenario.

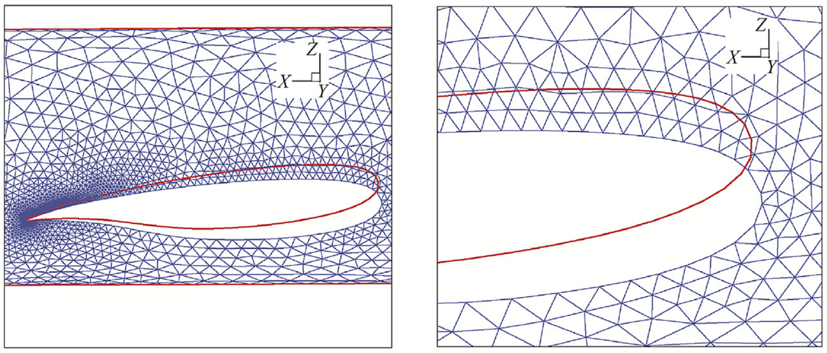


Fig. 15 Surface grid adaptation upon deformation of upper side of wing by using NURBS (left) and details near leading edge (right).

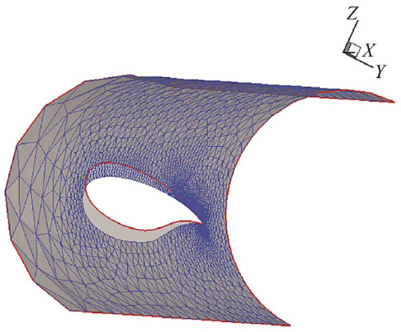


Fig. 16 Grid adaptation using NURBS based strategy has preserved underlying fuselage geometry.

Table 2 Mesh quality metrics comparison.

Item	Min quality	Max quality	Mean quality	Min dihedral
Original	0.345960	0.999750	0.841128	12.75480
Bump	0.281948	0.999750	0.840175	14.30680
Rotation	0.214193	0.999820	0.840795	6.729990
Slide	0.332527	0.999754	0.840888	15.58810

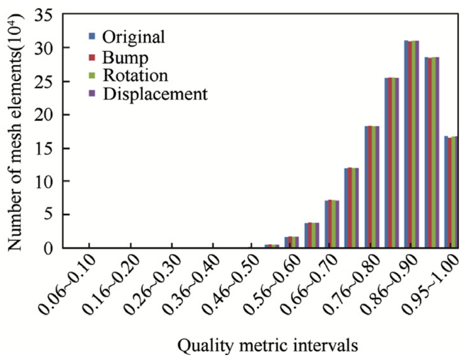


Fig. 17 Histogram of quality metric intervals with respect to number of mesh elements.

Table 3 Number of elements in each of grid quality intervals.

Interval	Original	Bump	Rotation	Slide
0–0.05	0	0	0	0
0.06–0.10	0	0	0	0
0.11–0.15	0	0	0	0
0.16–0.20	0	0	0	0
0.21–0.25	0	0	5	0
0.26–0.30	0	3	5	0
0.31–0.35	2	6	23	3
0.36–0.40	46	64	65	49
0.41–0.45	287	348	354	300
0.46–0.50	1415	1594	1473	1445
0.51–0.55	5490	5943	5639	5600
0.56–0.60	16791	17342	16972	16919
0.61–0.65	37583	38320	37784	37782
0.66–0.70	71120	72066	71465	71276
0.71–0.75	119721	120715	119996	119993
0.76–0.80	182178	183083	182332	182535
0.81–0.85	254927	255357	255191	254953
0.86–0.90	310805	309288	310117	310468
0.91–0.95	285834	284609	285659	285601
0.96–1.00	167650	165111	166769	166925

wing-pylon configurations, an efficient treatment of the intersections is required.

- A mesh deformation approach based on the underlying CAD geometry has been proposed to re-calculate the intersection.
- The proposed approach guarantees that the surface vertices are restricted to the baseline CAD.
- The proposed strategy is most useful when there are two NURBS panels.
- Limitations are detected when more than two NURBS surfaces are involved in the intersection. Since surface adaptation is restricted to the NURBS surface, in those situations with multiple panels, it is not clear how to propagate the deformation.

Further work will focus on the development of a strategy to propagate the surface deformation through several NURBS patches.

References

1. Braibant V, Fleury C. Shape optimal design using B-splines. *Comput Methods Appl Mech Eng* 1984;**44**(3):247–67.
2. Jameson A. Aerodynamic shape optimization using the adjoint method. Brussels: Aerodynamic Drag Prediction and Reduction; 2003. Report No.: VKI Lecture Series 2003-02.
3. Martin-Burgos MJ, Andrés-Pérez E, Widhalm M, Lozano Bitrian P. Non-uniform rational B-splines-based aerodynamic shape design optimization with the DLR TAU code. *Int J Aerosp Eng* 2012;**226**(10):1225–42.
4. Chiandussi G, Bugeda G, Oñate E. A simple method for automatic update of finite element meshes. *Int J Numer Methods Biomed Eng* 2000;**16**(1):1–19.
5. Rendall TCS, Allen CB. Unified fluid-structure interpolation and mesh motion using radial basis functions. *Int J Numer Meth Eng* 2008;**74**(10):1519–59.
6. Ying LM, Hewit WT. Point inversion and projection for NURBS curve and surface: control polygon approach. *Comput Aided Geom Des* 2003;**20**(2):79–99.
7. Sederberg TW, Finnigan GT, Li X, Lin H, Ipson H. Watertight trimmed NURBS. *ACM Trans Graph* 2008;**27**(3):1–8.
8. Gagnon H, Zingg DW. Geometry generation of complex unconventional aircraft with application to high-fidelity aerodynamic shape optimization. Reston: AIAA; 2013. Report No.: AIAA-2013-2850.
9. Hwang JT, Martins J. GeoMACH: Geometry-centric MDAO of aircraft configurations with high fidelity. Reston: AIAA; 2012. Report No.: AIAA-2012-5605.
10. Farhat C. Parallel and distributed solution of coupled non linear dynamic aeroelastic response problems. *Solving large-scale problems in mechanics: parallel and distributed computer applications*. Chichester: John Wiley and Sons Ltd; 1997. p. 243–301.
11. Johnson AA, Tezduyar TB. Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. *Comput Methods Appl Mech Eng* 1994;**119**(1–2):73–94.
12. Murayama M, Nakahashi K, Matsushima K. Unstructured dynamic mesh for large movement and deformation, Reston: AIAA; 2002. Report No.: AIAA-2002-0122.
13. Beckert A, Wendland H. Multivariate interpolation for fluid-structure-interaction problems using radial basis functions. *Aerosp Sci Technol* 2001;**5**(2):125–34.
14. Cordero-Garcia M, Gómez M, Ponsin J, Valero E. An interpolation tool for aerodynamic mesh deformation problems based on octree decomposition. *Aerosp Sci Technol* 2012;**23**(1):93–107.
15. Farhat C, Degand C, Koobus B, Lesoinne M. Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Comput Methods Appl Mech Eng* 1998;**163**(1):231–45.
16. Samareh JA. Aerodynamic shape optimization based on free-form deformation. Reston: AIAA; 2004. Report No.: AIAA-2004-4630.
17. Martin MJ, Andrés E, Lozano C, Valero E. Volumetric b-splines shape parameterization for aerodynamic shape optimization. *Aeroesp Sci Technol* 2014;**37**:26–36.
18. Selimovic I. Improved algorithms for the projection of points on NURBS curves and surfaces. *Comput Aided Geom D* 2006;**23**(5):439–45.
19. Sheng C, Allen CB. Efficient mesh deformation using radial basis functions on unstructured meshes. *AIAA J* 2013;**51**(3):707–20.
20. Piegl L, Tiller W. *The NURBS book*. Berlin Heidelberg: Springer-Verlag; 1995.
21. Boor C. *A practical guide to Splines*. Berlin Heidelberg: Springer-Verlag; 1978.
22. Technical Documentation of the DLR TAU-Code. Braunschweig: DLR institute of aerodynamics and flow technology; 2014.
23. Lozano C. Discrete surprises in the computation of sensitivities from boundary integrals in the continuous adjoint approach to inviscid aerodynamic shape optimization. *Comput Fluids* 2012;**56**:118–27.
24. AIAA. 3rd AIAA CFD drag prediction workshop. *2-Day workshop preceding the 25th APA conference*; 2006 June 3–4; San Francisco. Reston: AIAA; 2006.